Normalisasi Tabel Database

Kalamullah Ramli

Rationale

- Disain database yang baik akan menentukan kelangsungan aplikasi
- Common Problems:
  - Quick yielding paradigm..... ☹
  - Normalisasi tidak dipikirkan dengan matang, artinya program asal jadi dan asal bisa dipakai...
  - Database tidak jalan, susah di *maintain*, sukar diwariskan, membingungkan
- NORMALISASI harus dilakukan agar program dan data kita dipakai dalam jangka waktu yang tidak terbatas

Keunggulan dan Kekurangan

- Memecah Tabel (sangat) kompleks menjadi beberapa tabel Sederhana
  - Table Master (*input once, constant at most of the time*)
  - Table Transaksi (*dynamic, related to Table Master*)
- Benefit?
  - *Easier to manage, to maintain, to report*
  - *Improves data integrity*
  - *Table reusable*
    - Ketika terjadi perubahan atas tabel master, maka tabel transaksi otomatis juga berubah menyesuaikan tabel master

Keunggulan dan Kekurangan

- Kerugian?
  - Increases number of JOIN
  - Improves Query Complexity
  - In few cases, affecting performance
- Apa yang perlu dipersiapkan?
  - Persiapan yang matang (ER Diagram, UML)
  - Bagaimanapun juga aplikasi apapun yang kita buat, normalisasi jangan ditinggalkan

Kapan Diperlukan?

- Normalisasi database biasanya jarang dilakukan dalam database skala kecil, dan dianggap tidak diperlukan pada penggunaan personal
- Namun seiring dengan berkembangnya informasi yang dikandung dalam sebuah database, proses normalisasi akan sangat membantu dalam menghemat ruang yang digunakan oleh setiap tabel di dalamnya, sekaligus mempercepat proses permintaan data

Proses Normalisasi

- Proses normalisasi model data dapat diringkas sebagai berikut:
  - Menemukan entitas-entitas utama dalam model data
  - Menemukan hubungan antara setiap entitas
  - Menentukan atribut yang dimiliki masing-masing entitas

Langkah-langkah Normalisasi (1 of 3)

- Bentuk Normal Pertama (1NF)
  - Sebuah model data dikatakan memenuhi bentuk normal pertama apabila setiap atribut yang dimilikinya memiliki satu dan hanya satu nilai.
  - Apabila ada atribut yang memiliki nilai lebih dari satu, atribut tersebut adalah kandidat untuk menjadi entitas tersendiri

Ex 1. Unnormalized to 1NF

STUDI KASUS TOKO ABC

No. Faktur :
Tanggal : Kepada :

No.	Nama	Jumlah	Harga	Total
			Total Bayar	
			Diskon	
			Jumlah Bayar	

Petugas : .....

Ex 1. Unnormalized to 1NF

1. Tabel yang memiliki field dengan banyak data / tidak tunggal

No_Faktur	Tanggal	Nama_pelanggan	Daftar_Belanja
05070101	29/05/07	Pitoyo	Bedak, Beras, Minyak Tanah, Buku
05070102	29/05/07	Bowo	Baby Oil, Garam, Gula, Pensil
05070103	30/05/07	Erlina	Sikat gigi, Sabun, Odol, Sampo
06070001	01/06/07	Dayat	Beras

2, Tabel dengan field yang mengalami repeating groups

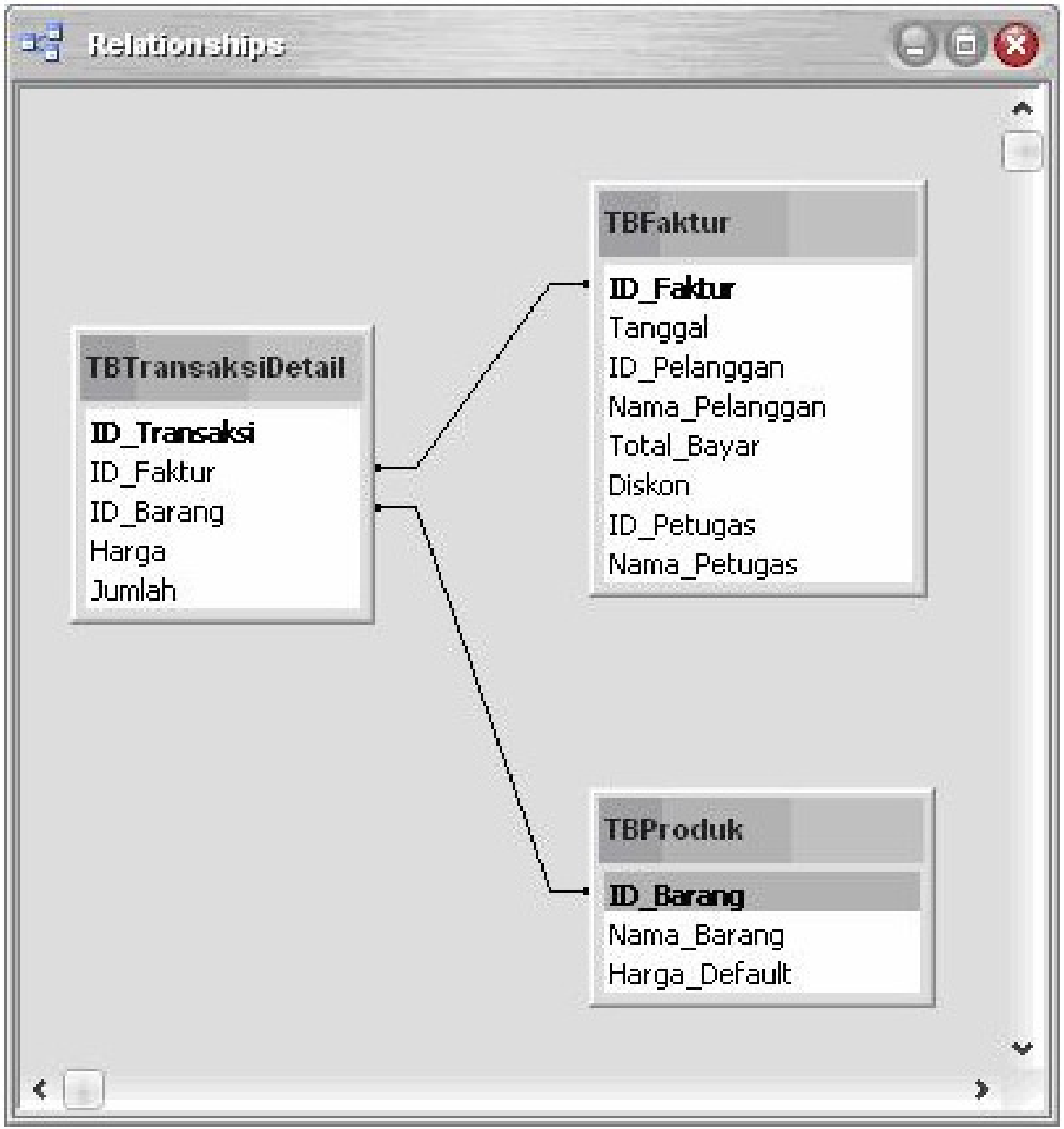
No_Faktur	Tanggal	Pelanggan	Belanja 1	Harga1	Belanja2	Harga2	Belanja3	Har
05070101	29/05/07	Pitoyo	Bedak	1500	Beras	10000	Minyak Tanah	350
05070102	29/05/07	Bowo	Baby Oil	5600	Garam	2500	Gula	400
05070103	30/05/07	Erlina	Sikat gigi	12000	Sabun	2500	Odol	130
06070001	01/06/07	Dayat	Beras	25000				

Ex 1. Unnormalized to 1NF

- Tabel di atas menggambarkan bentuk basis data yang belum ternormalisasi, karena suatu relasi memenuhi 1-NF jika dan hanya jika setiap atribut hanya memiliki nilai tunggal dalam satu baris / record dan tidak mengalami repeating groups
- Implementasi 1-NF dari table data yang belum ternormalisasi di atas adalah dengan cara mengeliminasi keberadaan repeating groups dan dekomposisi relasi menjadi dua atau lebih dengan syarat "tidak boleh ada informasi yang hilang karena proses dekomposisi"

Ex 1. Unnormalized to 1NF

1. Membuat 3 tabel yang memiliki fungsi sbb:
   - **TBFaktur**, berfungsi untuk menyediakan atribut-atribut yang bersifat atomic dari tiap nomor faktur (ID_Faktur), seperti : Tanggal, Nama_Pelanggan, Total_Bayar, Diskon dan Nama_Petugas
   - **TBProduk**, berfungsi untuk menyediakan atribut-atribut yang berulang atau tidak bernilai tunggal pada tiap nomor faktur (ID_Faktur), seperti : Nama_Barang dan harga
   - **TBTransaksiDetail**, berfungsi sebagai penghubung antara nomor faktur (ID_Faktur) dengan kode barang (ID_Barang) agar proses dekomposisi tidak menyebabkan kerusakan informasi.



o 1NF

ɔut dan
ai berikut

Ex 1. Unnormalized to 1NF

3. Pada table **TBTransaksiDetail** terdapat atribut **“Harga”** yang berfungsi untuk menyimpan harga per transaksi, sedangkan atribut **“Harga_Default”** yang terdapat pada table TBProduk adalah atribut yang berfungsi untuk menyimpan harga barang terbaru dari tiap jenis barang
   - Hal ini berguna untuk mengantisipasi adanya perubahan harga dari waktu ke waktu
4. Primary key yang digunakan pada TBTransaksiDetail adalah **“ID_Transaksi”**
   - Atribut kunci tersebut merupakan candidate key yang dibentuk dari superkey hasil penggabungan 2 atribut yaitu : ID_Faktur dan ID_Barang

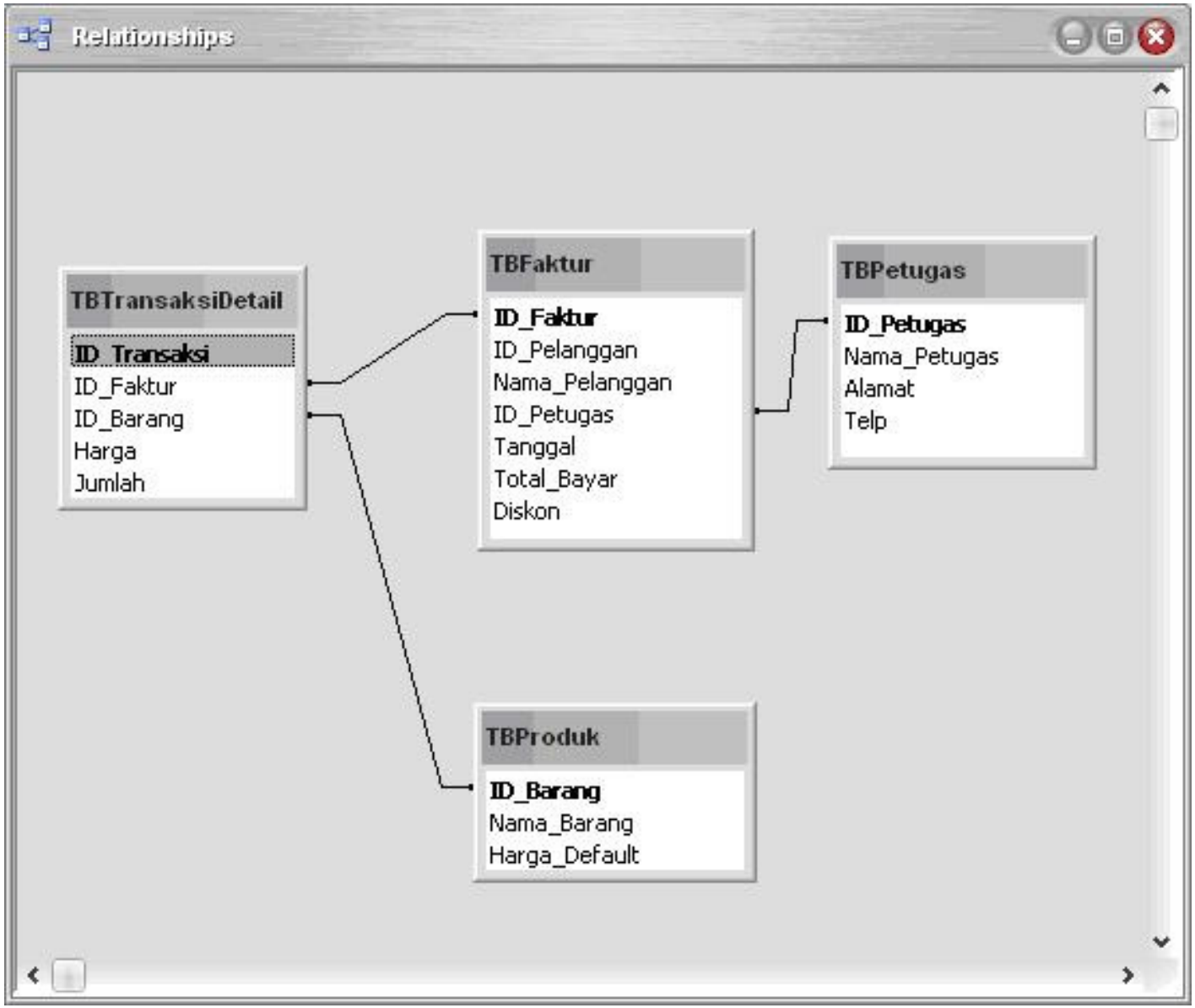
Langkah-langkah Normalisasi (2 of 3)

- Suatu relasi berada dalam Bentuk Normal Kedua (2NF) jika dan hanya jika:
  - Berada dalam Bentuk Normal Kesatu(1NF)
  - Semua atribut bukan kunci memiliki dependensi sepenuhnya dengan kunci primer (Primary Key)

Ex 2. 1NF to 2NF

- Jika kita lihat kembali relasi bentuk 1-NF di atas, maka atribut bukan kunci pada table TBFaktur yang tidak memiliki dependensi sepenuhnya dengan primary key (ID_Faktur), yaitu: “Nama_Petugas”
  - Mengeliminasi atribut “Nama_Petugas” dari table TBFaktur
  - Membuat tabel **TBPetugas**, menyediakan atribut-atribut yang terkait dengan identitas dan data pelanggan

Ex 2. 1NF to 2NF



Langkah-langkah Normalisasi (3 of 3)

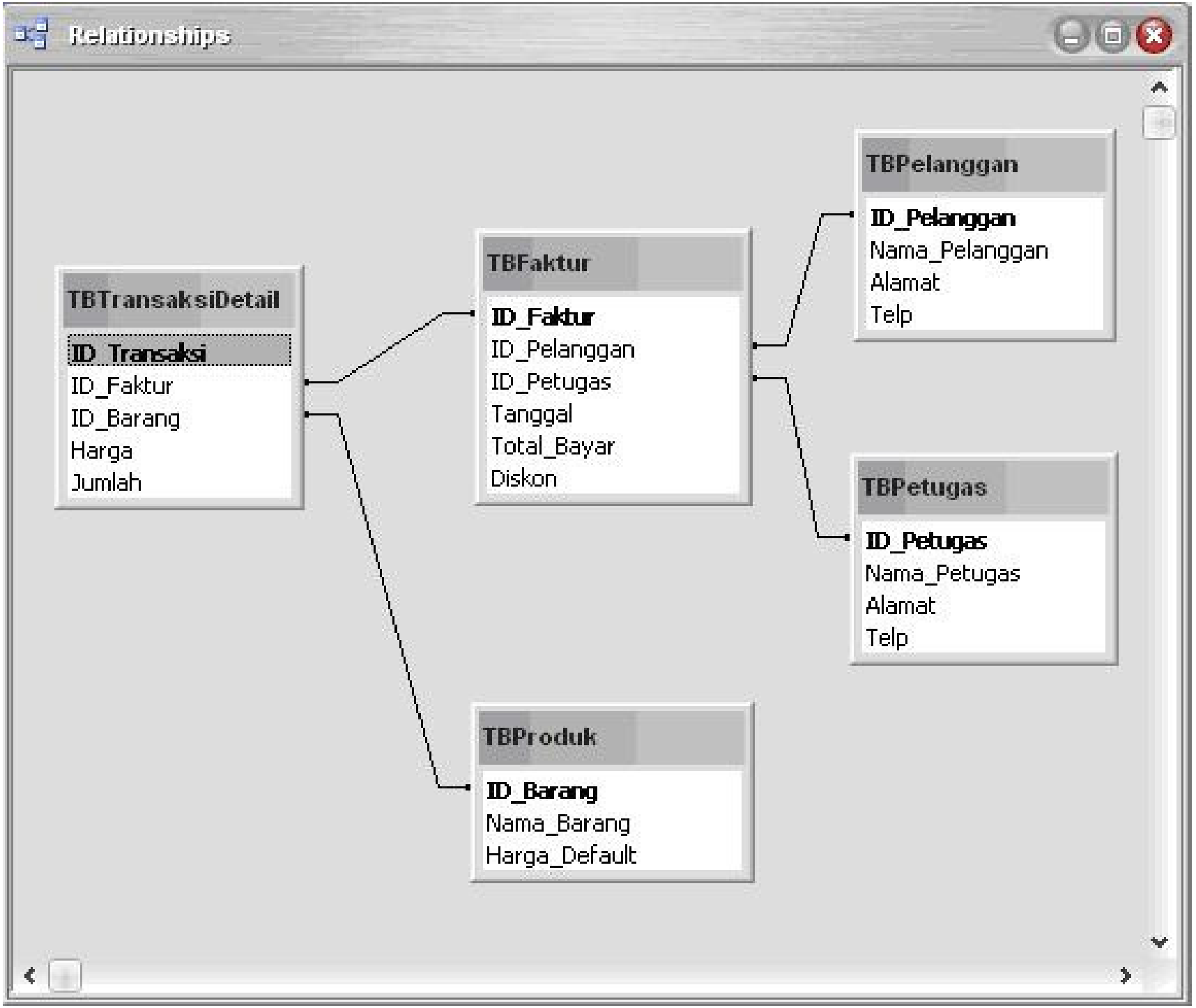
- Suatu Model berada dalam Bentuk Normal Ketiga (3NF) jika dan hanya jika:
  - memenuhi bentuk normal kedua
  - tidak ada satupun atribut non-*identifying (bukan pengidentifikasi unik) yang* bergantung pada atribut non-*identifying lain*
    - *Apabila ada, pisahkan salah satu atribut tersebut* menjadi entitas baru, dan atribut yang bergantung padanya menjadi atribut entitas baru tersebut

Ex 3. 2NF to 3NF

- Pada Bentuk Normal Kedua (2NF) atribut yang terkait dengan “Nama_Pelanggan” tidak didekomposisi dari table TBFaktur karena atribut tersebut masih memiliki dependensi fungsional dengan primary key (ID_Faktur) karena tiap nomor faktur akan berbeda untuk tiap pembeli/pelanggan
- Tetapi pada tahap 3-NF (Third Normal Form), atribut “Nama_Pelanggan” harus didekomposisi relasi karena pada tahap ini atribut bukan kunci tidak boleh ada yang berdependensi transitif dengan kunci primer

Ex 3. 2NF to 3NF

- Atribut “Nama_Pelanggan” dikatakan berdependensi transitif terhadap primary key (ID_Faktur) karena :
  - ID_Pelanggan → Nama_Pelanggan (Nama_Pelanggan berdependensi fungsional terhadap ID_Pelanggan)
  - ID_Faktur → ID_Pelanggan (ID_Pelanggan berdependensi fungsional terhadap ID_Faktur, karena tiap nomor faktur akan dikeluarkan untuk suatu ID_Pelanggan tertentu)
  - Sehingga dikatakan bahwa ID_Faktur memiliki dependensi transitif terhadap atribut Nama_Pelanggan


Relationships
TBTransaksiDetail
ID_Transaksi
ID_Faktur
ID_Barang
Harga
Jumlah
TBFaktur
ID_Faktur
ID_Pelanggan
ID_Petugas
Tanggal
Total_Bayar
Diskon
TBPelanggan
ID_Pelanggan
Nama_Pelanggan
Alamat
Telp
TBPetugas
ID_Petugas
Nama_Petugas
Alamat
Telp
TBProduk
ID_Barang
Nama_Barang
Harga_Default